Question1

***Solution*:**

1. Setup

Let Exp be the original expression, $Exp[i:j]$ is $Exp's$ sub expression from index i to j. Denote $T(i,j)$ is the number of ways one can put parentheses in the expression $Exp[i:j]$ such that it will evaluate to true and the $F(i,j)$ is the number of ways one can put parentheses in the expression $Exp[i:j]$ such that it will evaluate to false.

2. Subproblems

The subproblems is 'Count the number of ways one can put parentheses in the expression such that it will evaluate to true of $Exp[i:k]$ and $Exp[k+1:j]$ separately'. Hence we need to count $T(i,k), F(i,k), T(k+1,j), F(k+1,j)\ for\ i \leq k < j$.

If we have solved all the subproblems. Let OP be the operator between $Exp[k]\ and\ Exp[k+1]$, then we can get these:

(1) $\boldsymbol{if\ OP\ =\ AND}$:

$\boldsymbol{AND}$	$\boldsymbol{true}$	$\boldsymbol{false}$
$true$	$true$	$false$
$false$	$false$	$false$

So only the $Exp[i:k]$ and $Exp[k+1:j]$ are $True$, $Exp[i:j]$ can be $True$. Otherwise, $Exp[i:j]$ is $False$. So we can get $T(i,j)\ and\ F(i,j)$:

$$T(i,j) = \sum_{k=i}^{j-1} T(i,k) * T(k+1,j)$$

$$F(i,j) = \sum_{k=i}^{j-1}(F(i,k) * F(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j))$$

(2) $\boldsymbol{if\ OP\ =\ OR}$:

$\boldsymbol{OR}$	$\boldsymbol{true}$	$\boldsymbol{false}$
$true$	$true$	$true$
$false$	$true$	$false$

So only the $Exp[i:k]$ and $Exp[k+1:j]$ are $False$, $Exp[i:j]$ can be $False$.

Otherwise, $Exp[i:j]$ is $True$. So we can get $T(i,j)\ and\ F(i,j)$:

$$T(i,j) = \sum_{k=i}^{j-1}(T(i,k) * T(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j))$$

$$F(i,j) = \sum_{k=i}^{j-1}(F(i,k) * F(k+1,j))$$

(3) $\boldsymbol{if\ OP\ =\ NAND}$:

$\boldsymbol{NAND}$	$\boldsymbol{true}$	$\boldsymbol{false}$
$true$	$false$	$true$
$false$	$true$	$true$

So only the $Exp[i:k]$ and $Exp[k+1:j]$ are $true$, $Exp[i:j]$ can be $False$.

Otherwise, $Exp[i:j]$ is $True$. So we can get $T(i,j)\ and\ F(i,j)$:

$$T(i,j) = \sum_{k=i}^{j-1}(F(i,k) * F(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j))$$

$$F(i,j) = \sum_{k=i}^{j-1}(T(i,k) * T(k+1,j))$$

(4) $\boldsymbol{if\ OP\ =\ NOR}$:

$\boldsymbol{NAND}$	$\boldsymbol{true}$	$\boldsymbol{false}$
$true$	$false$	$false$
$false$	$false$	$true$

So only the $Exp[i:k]$ and $Exp[k+1:j]$ are $true$, $Exp[i:j]$ can be $False$. Otherwise, $Exp[i:j]$ is $True$. So we can get $T(i,j)\ and\ F(i,j)$:

$$T(i,j) = \sum_{k=i}^{j-1}(F(i,k) * F(k+1,j))$$

$$F(i,j) = \sum_{k=i}^{j-1}(T(i,k) * T(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j))$$

3. Build-up order

Solve the subproblems in the order $T(1,1), F(1,1), T(2,2), F(2,2), T(1,2), F(1,2), T(3,3), F(3,3), T(2,3), F(2,3), T(1,3), F(1,3), \dots, F(1,n)$

4. base case

$$\begin{cases} If\ Exp[i]\ is\ True : T(i:i) = 1\ and\ F(i:i) = 0 \\ If\ Exp[i]\ is\ False : T(i:i) = 0\ and\ F(i:i) = 1 \end{cases}$$

5. Recursion

$T(i,j) =$

$$\sum_{k=i}^{j-1}\begin{cases} T(i,k) * T(k+1,j)\ ;if\ OP = AND \\ T(i,k) * T(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j)\ ;if\ OP = OR \\ F(i,k) * F(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j)\ ;if\ OP = NAND \\ F(i,k) * F(k+1,j)\ ;if\ OP = NOR \end{cases}$$

$F(i,j) =$

$$\sum_{k=i}^{j-1}\begin{cases} F(i,k) * F(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j)\ ;if\ OP = AND \\ F(i,k) * F(k+1,j)\ ;if\ OP = OR \\ T(i,k) * T(k+1,j)\ ;if\ OP = NAND \\ T(i,k) * T(k+1,j) + F(i,k) * T(k+1,j) + T(i,k) * F(k+1,j);if\ OP = NOR \end{cases}$$

6. Final solution

$$number\ of\ ways = T(1, n)$$

We can solve this problem by filling two tables in the top to the bottom manner just like below and get the answer at the last blank $\boldsymbol{T}(1, n)$ on the True table.

$\boldsymbol{T(i,j)}$	$\boldsymbol{i=1}$	$\boldsymbol{i=2}$	…	$\boldsymbol{i=n}$
$j=1$				
$j=2$				
…				
$j=n$				

$\boldsymbol{F(i,j)}$	$\boldsymbol{i=1}$	$\boldsymbol{i=2}$	…	$\boldsymbol{i=n}$
$j=1$				
$j=2$				
…				
$j=n$				

7. Time complexity

There are $2n^2$ subproblems and each subproblems is $O(n)$ hence the overall time complexity of the algorithm is $O(n^2) * O(n) = O(n^3)$.